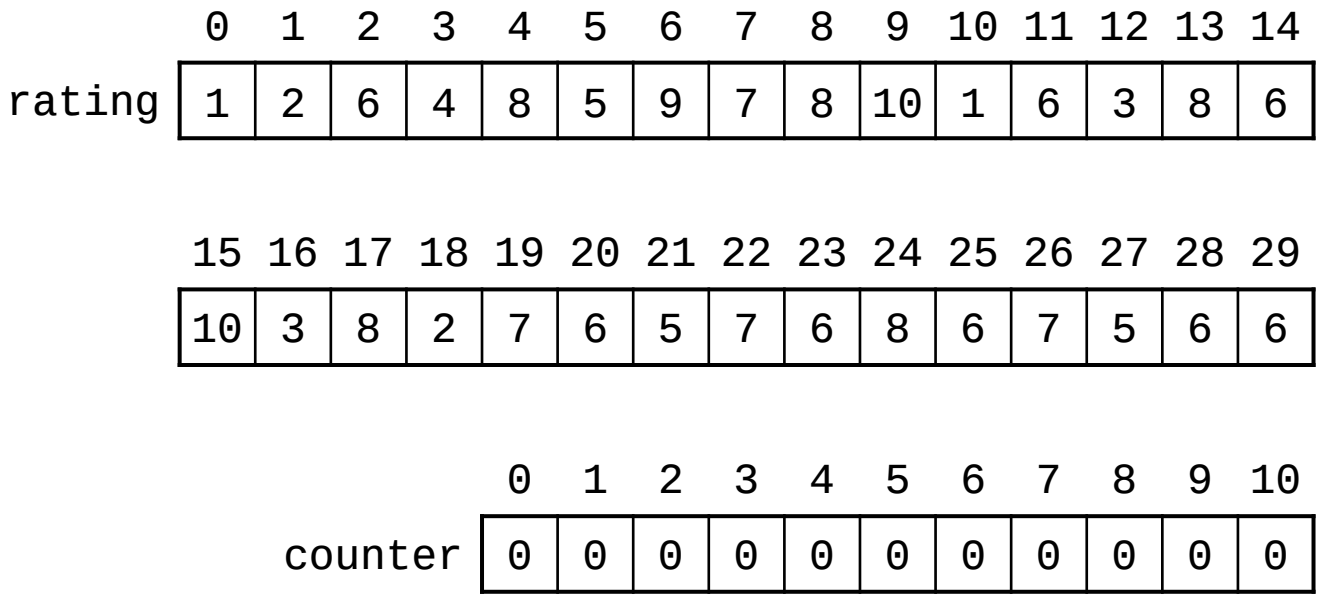
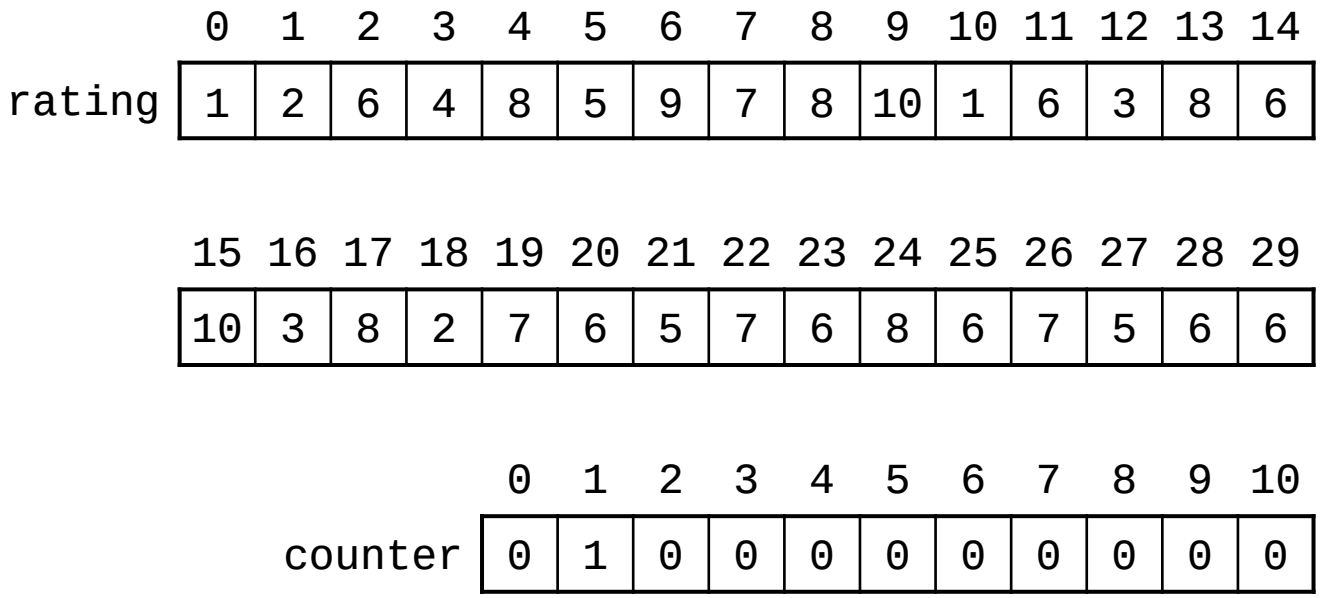
Arrays and Vectors

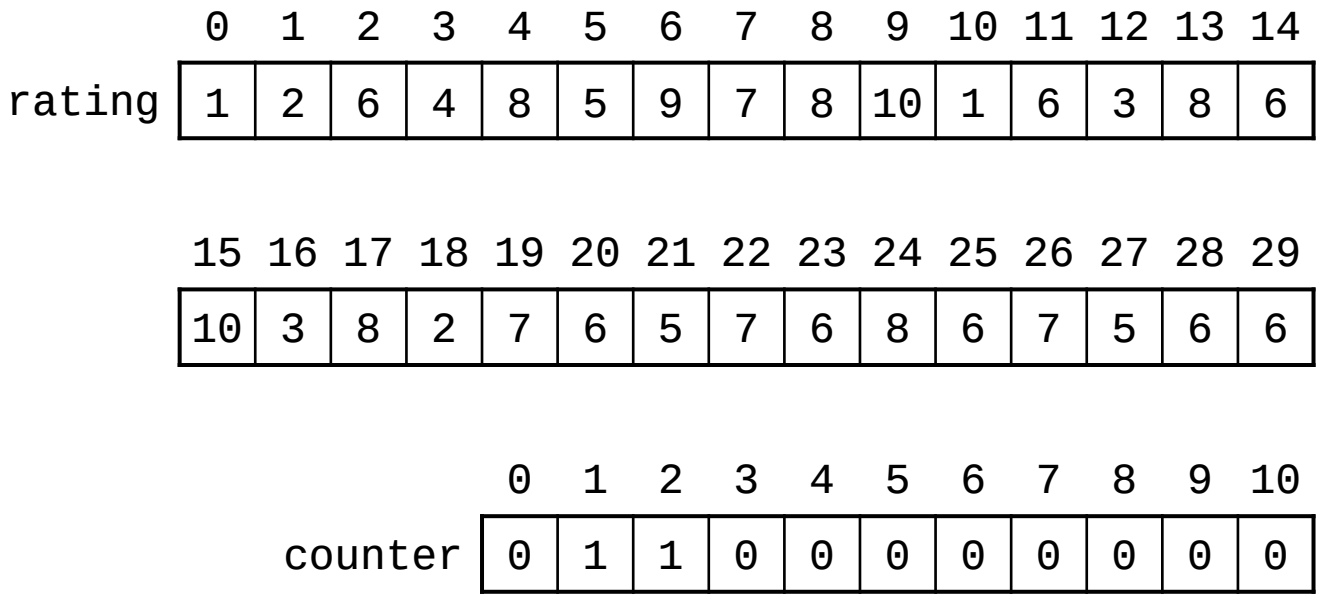
```cpp
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
   const int ratingSize = 40;
   const int counterSize = 11;

   const int rating[ ratingSize ] = { 1, 2, 6, 4, 8, 5, 9, 7, 8,
      10, 1, 6, 3, 8, 6, 10, 3, 8, 2, 7, 6, 5, 7, 6, 8, 6, 7,
      5, 6, 6, 5, 6, 7, 5, 6, 4, 8, 6, 8, 10 };

   int counter[ counterSize ] = {};

   for( int i = 0; i < ratingSize; i++ )
      counter[ rating[ i ] ]++;

   cout << "Rating" << setw( 9 ) << "Counter" << endl;

   for( int i = 1; i < counterSize; i++ )
      cout << setw( 6 ) << i << setw( 9 ) << counter[ i ]  << endl;
}
```



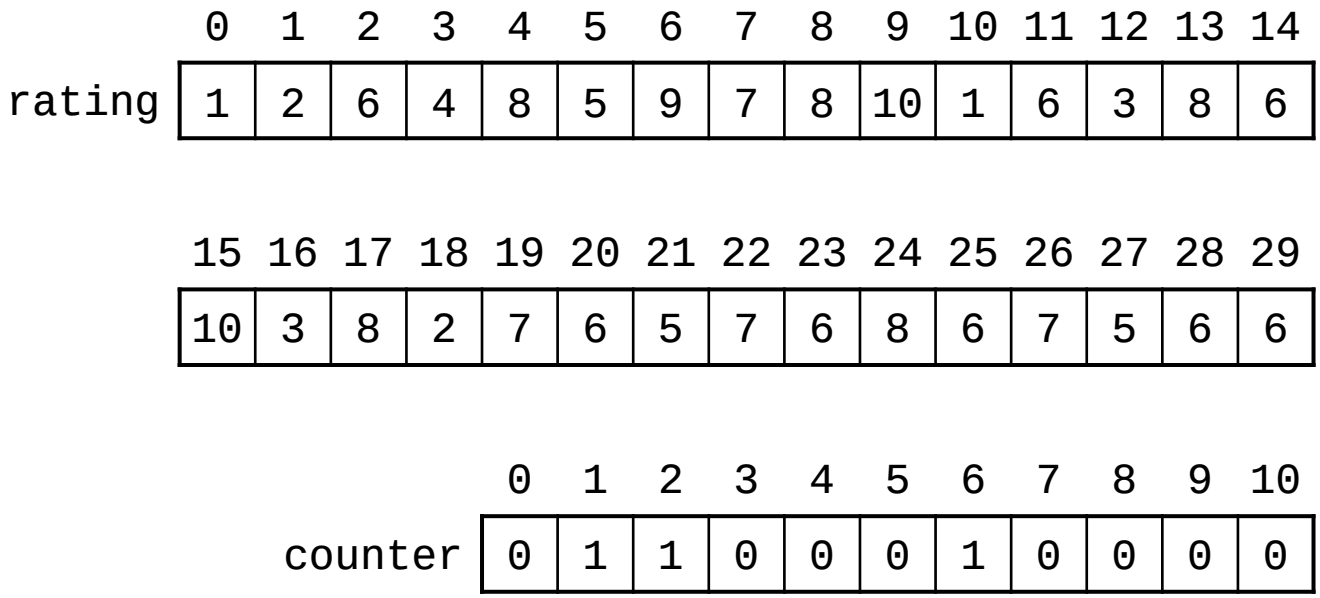

```
for( int i = 0; i < 30; i++ )
   counter[ rating[ i ] ]++;
```



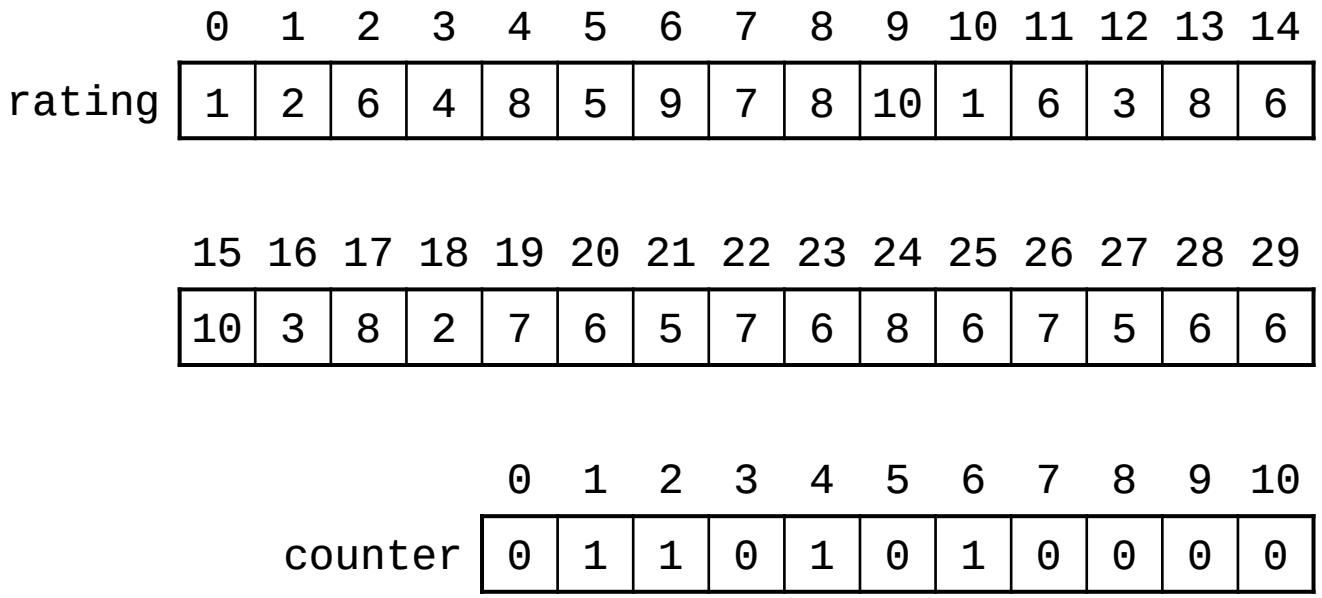

```
for( int i = 0; i < 30; i++ )
   counter[ rating[ i ] ]++;
```



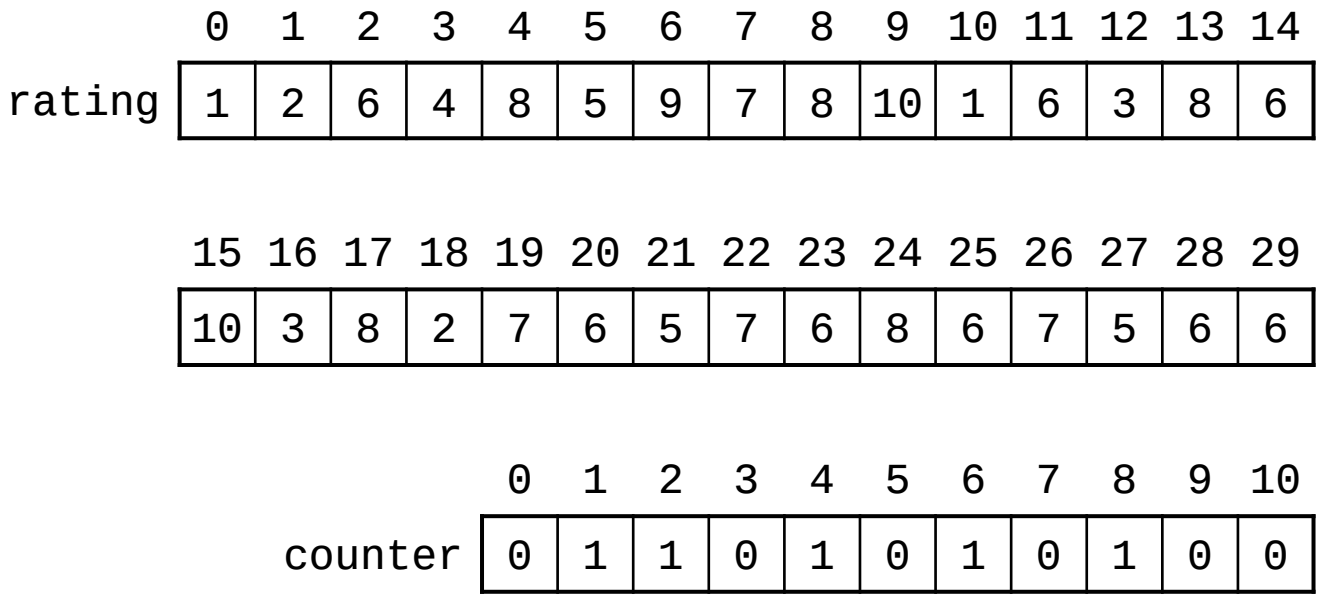

```
for( int i = 0; i < 30; i++ )
   counter[ rating[ i ] ]++;
```



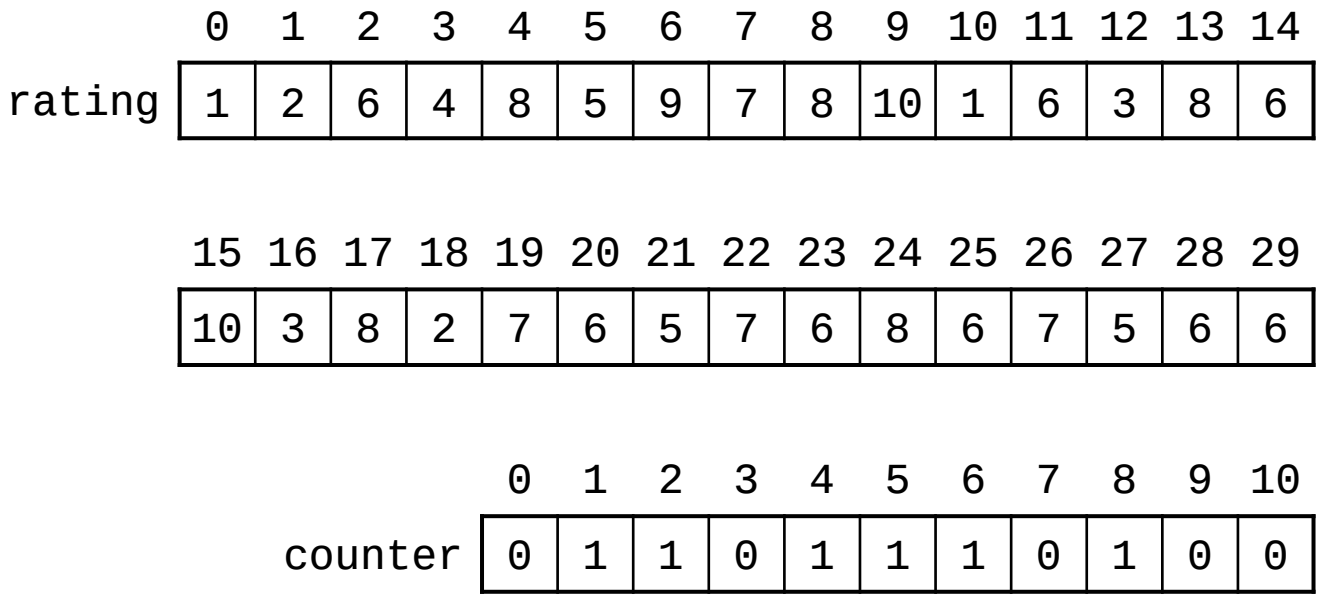

```
for( int i = 0; i < 30; i++ )
   counter[ rating[ i ] ]++;
```



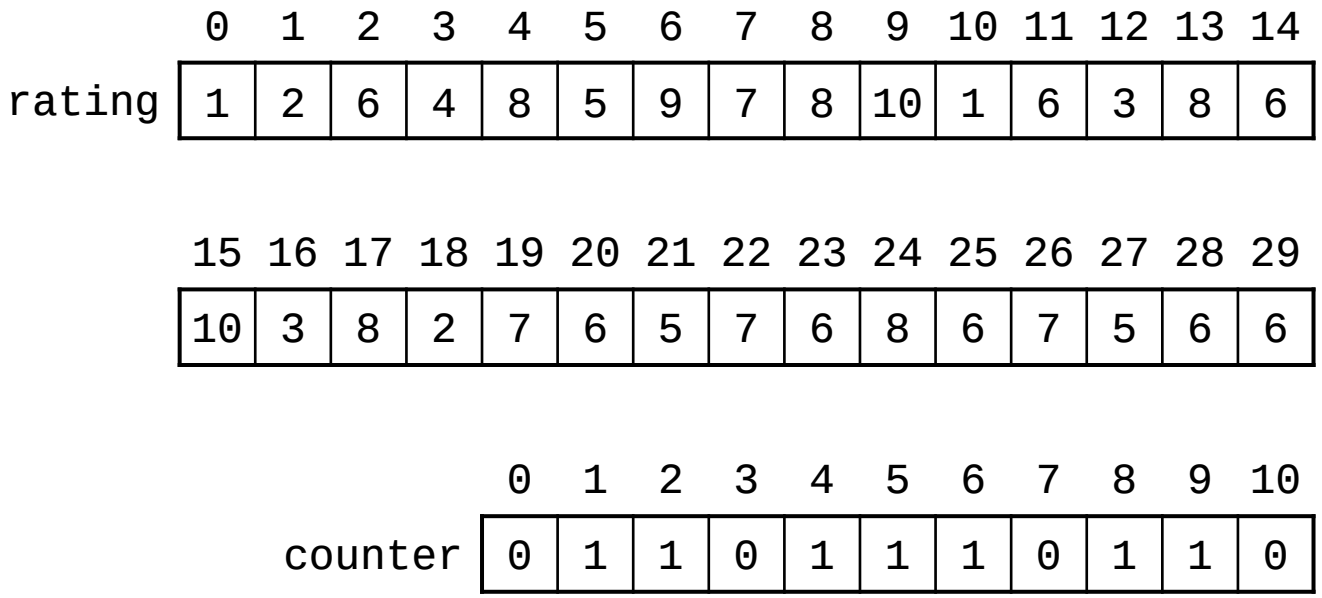

```
for( int i = 0; i < 30; i++ )
   counter[ rating[ i ] ]++;
```



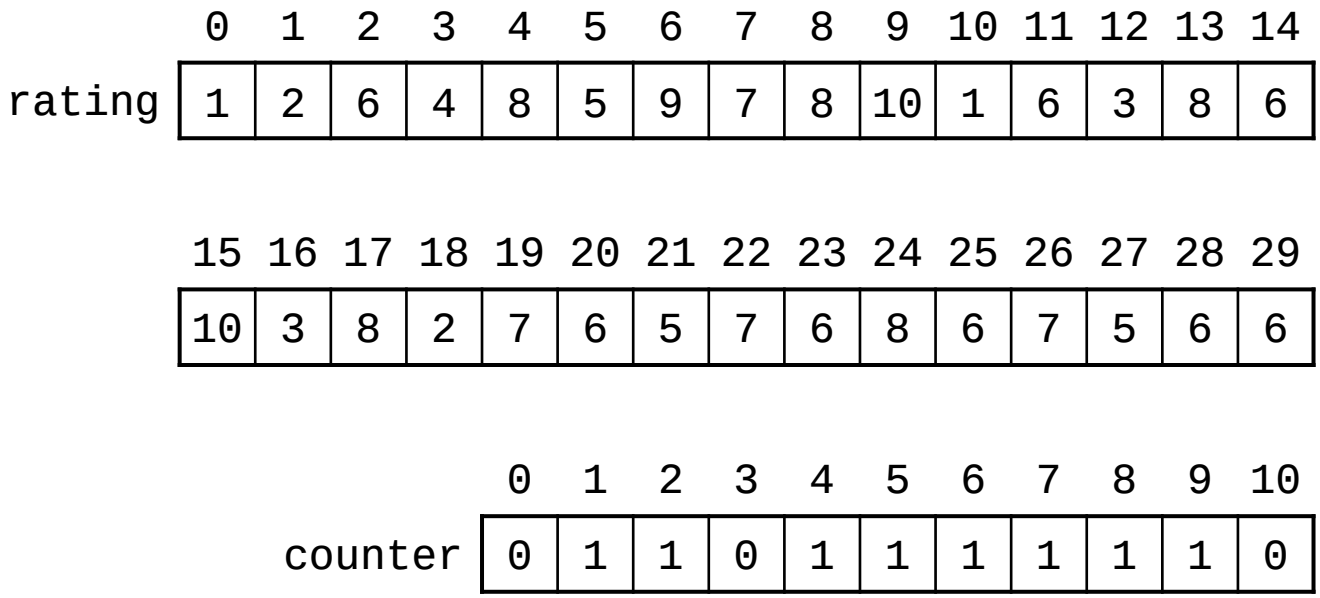

```
for( int i = 0; i < 30; i++ )
   counter[ rating[ i ] ]++;
```



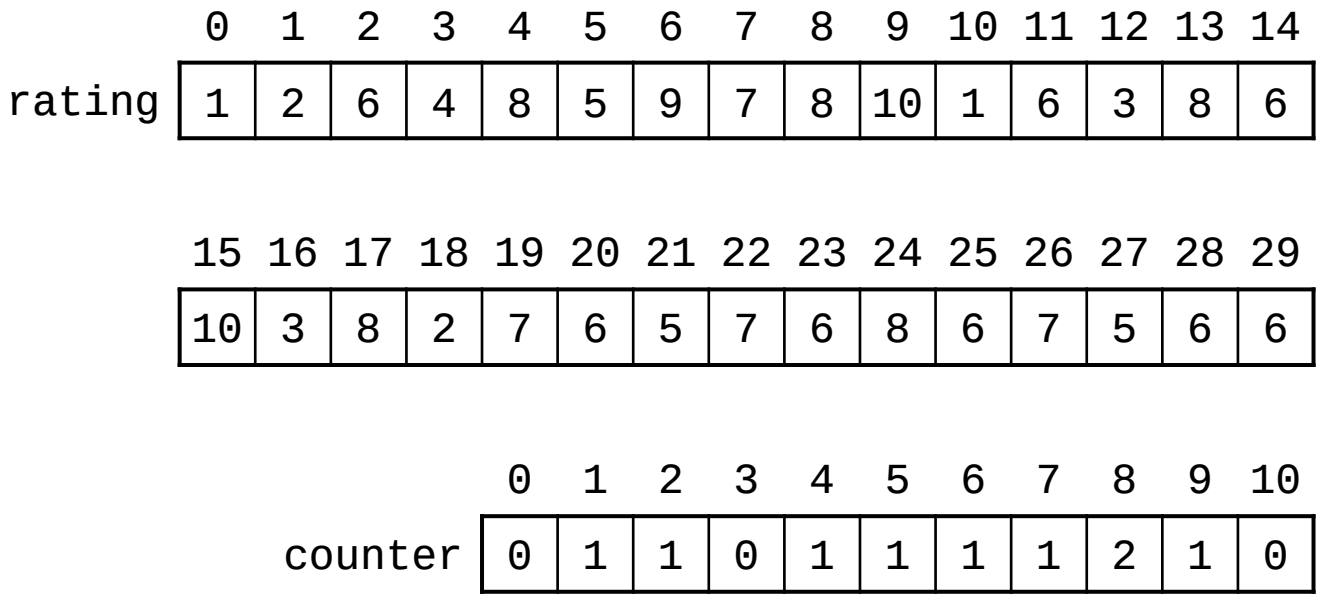

```
for( int i = 0; i < 30; i++ )
   counter[ rating[ i ] ]++;
```



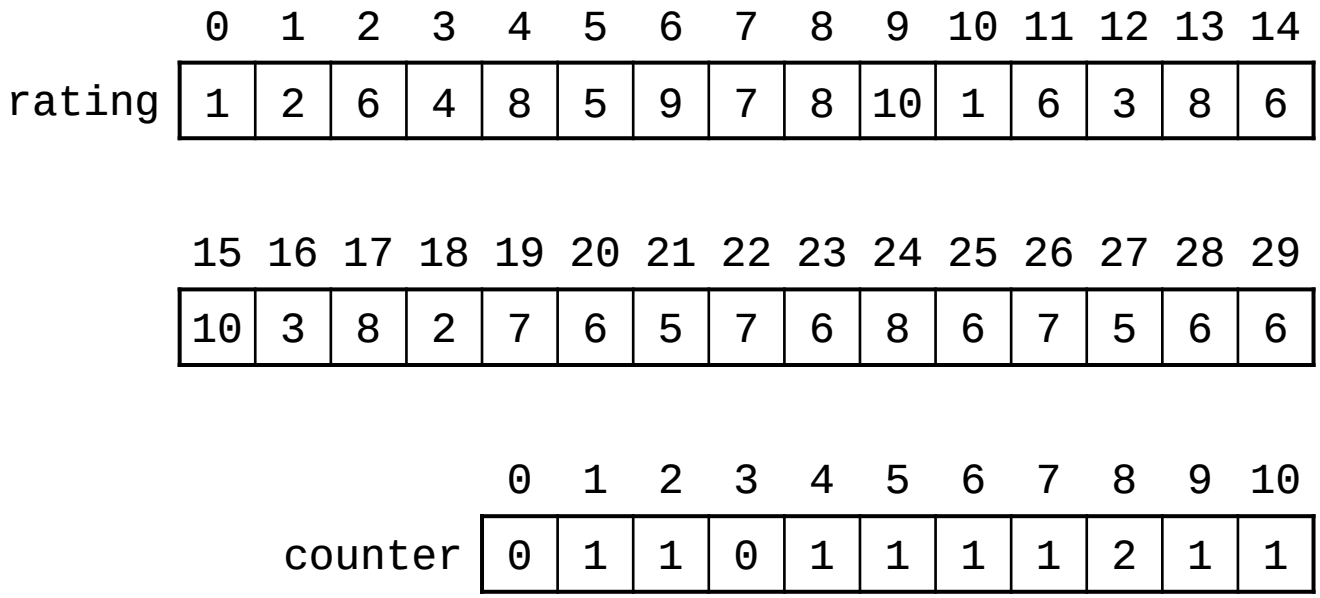

```
for( int i = 0; i < 30; i++ )
   counter[ rating[ i ] ]++;
```



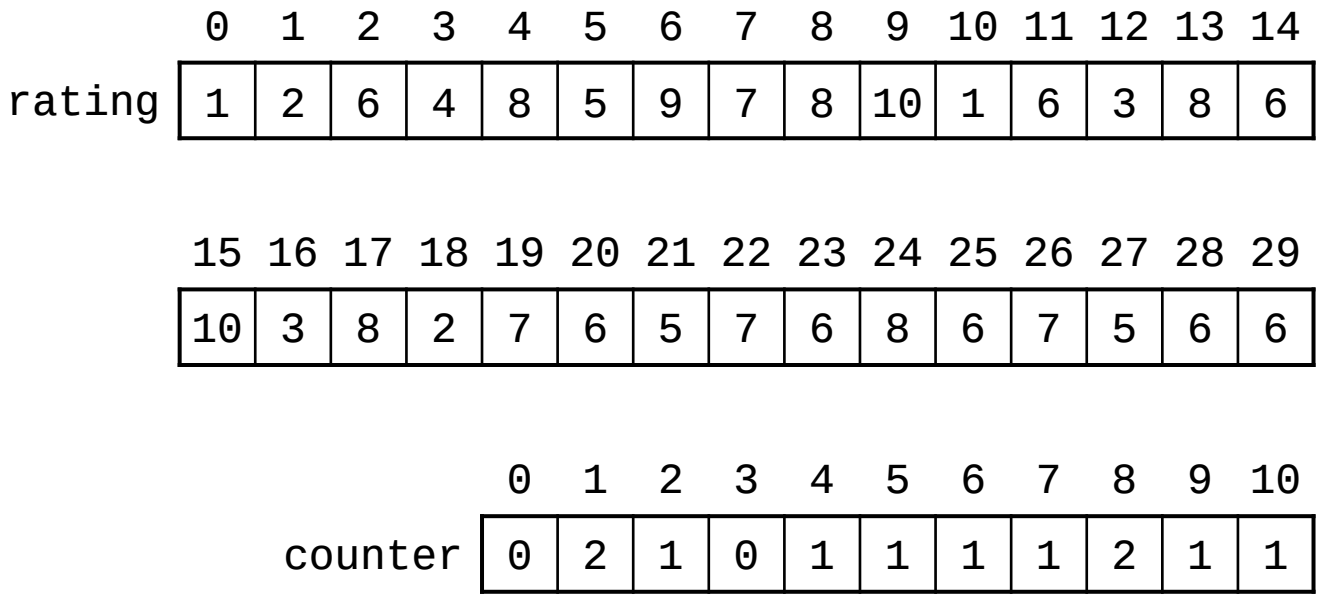

```
for( int i = 0; i < 30; i++ )
   counter[ rating[ i ] ]++;
```




```
for( int i = 0; i < 30; i++ )
   counter[ rating[ i ] ]++;
```




```
for( int i = 0; i < 30; i++ )
   counter[ rating[ i ] ]++;
```




```
for( int i = 0; i < 30; i++ )
   counter[ rating[ i ] ]++;
```

Rating	Counter
1	2
2	2
3	2
4	2
5	5
6	11
7	5
8	7
9	1
10	3

```
// Poll analysis program.
#include <iostream>
using std::cout;
using std::endl;
#include <iomanip>
using std::setw;

int main()
{
   const int ratingSize = 40;
   const int counterSize = 11;
   const int rating[ ratingSize ] = { 1, 2, 6, 4, 8, 5, 9,
       7, 8, 10, 1, 6, 3, 8, 6, 10, 3, 8, 2, 7, 6, 5, 7, 6,
       8, 6, 7, 5, 6, 6, 5, 6, 7, 5, 6, 4, 8, 6, 8, 10 };
   int counter[ counterSize ] = {};
```

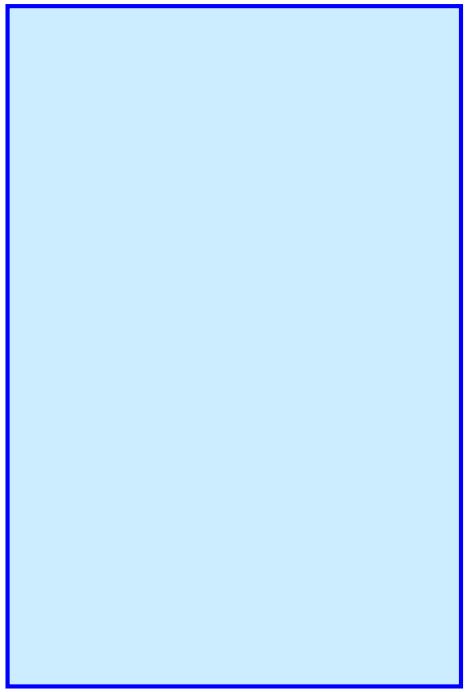
```
for( int i = 0; i < ratingSize; i++ )
   switch( rating[ i ] )
   {
      case 1:
         ++counter[ 1 ];  break;
      case 2:
         ++counter[ 2 ];  break;
      case 3:
         ++counter[ 3 ];  break;
      case 4:
         ++counter[ 4 ];  break;
      case 5:
         ++counter[ 5 ];  break;
      case 6:
         ++counter[ 6 ];  break;
      case 7:
         ++counter[ 7 ];  break;
      case 8:
         ++counter[ 8 ];  break;
      case 9:
         ++counter[ 9 ];  break;
      case 10:
         ++counter[ 10 ]; break;
      default:
         cout << "program should never get here!";
   }
```

```
for( int i = 0; i < ratingSize; i++ )
   switch( rating[ i ] )
   {
      case 1:
         ++counter[ rating[ i ] ];  break;
      case 2:
         ++counter[ rating[ i ] ];  break;
      case 3:
         ++counter[ rating[ i ] ];  break;
      case 4:
         ++counter[ rating[ i ] ];  break;
      case 5:
         ++counter[ rating[ i ] ];  break;
      case 6:
         ++counter[ rating[ i ] ];  break;
      case 7:
         ++counter[ rating[ i ] ];  break;
      case 8:
         ++counter[ rating[ i ] ];  break;
      case 9:
         ++counter[ rating[ i ] ];  break;
      case 10:
         ++counter[ rating[ i ] ];  break;
      default:
         cout << "program should never get here!";
   }
```

```
for( int i = 0; i < ratingSize; i++ )
   ++counter[ rating[ i ] ];
```

```
   cout << "Rating" << setw( 9 ) << "Counter" << endl;

   for( int i = 1; i < counterSize; i++ )
      cout << setw( 6 ) << i
           << setw( 9 ) << counter[ i ] << endl;
}
```

```
int number = 1; // global variable
int main()
{
   cout << number << endl;
   int number = 3; // local variable to main
   cout << number << endl;
   {
      cout << number << endl;
      int number = 7;
      cout << number << endl;
   }
   cout << number << endl;
   useGlobal();
   cout << number << endl;
   cout << ::number << endl;
}

void useGlobal()
{
   cout << number << endl;
   number++;
   cout << number << endl;
}
```



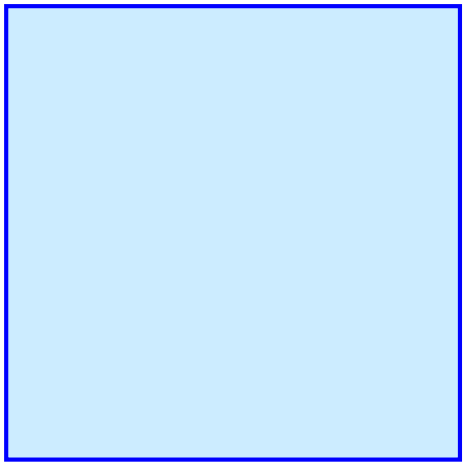
```
int number = 1; // global variable
int main()
{
   cout << number << endl;
   int number = 3; // local variable to main
   cout << number << endl;
   {
      cout << number << endl;
      int number = 7;
      cout << number << endl;
   }
   cout << number << endl;
   useGlobal();
   cout << number << endl;
   cout << ::number << endl;
}

void useGlobal()
{
   cout << number << endl;
   number++;
   cout << number << endl;
}
```

```
1
3
3
7
3
1
2
3
2
```

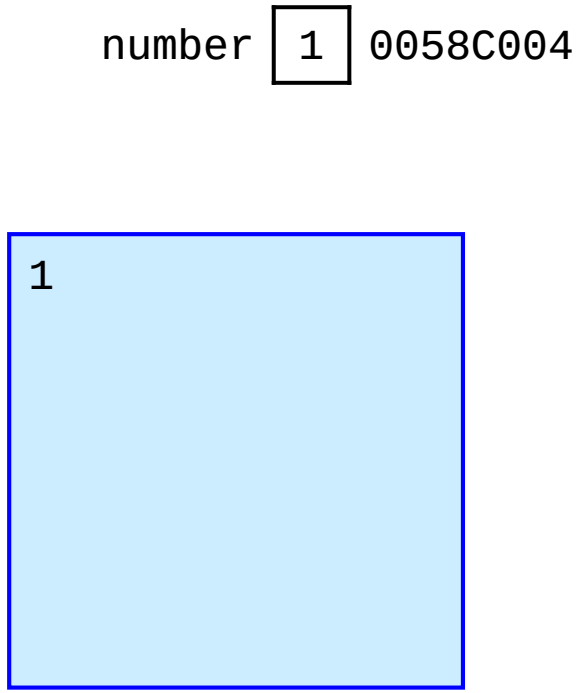
```
void useStatic();

int main()
{
   useStatic();
   useStatic();
}

// function to demonstrate a static local variable
void useStatic()
{
   // initialized first time useStatic is called
   static int number = 1;

   cout << number << endl;
   number++;
   cout << number << endl;
}
```

```
int main()
{
   useStatic();
   useStatic();
}

void useStatic()
{
   static int number = 1;
   cout << number << endl;
   number++;
   cout << number << endl;
}
```

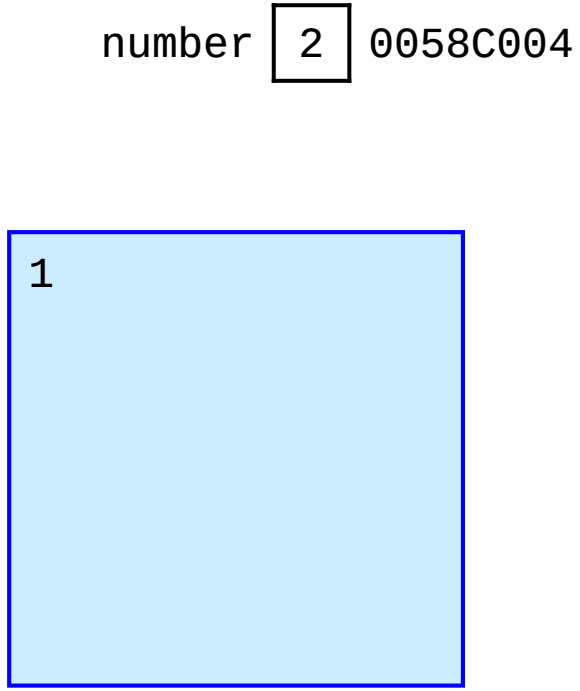
number | 1 | 0058C004



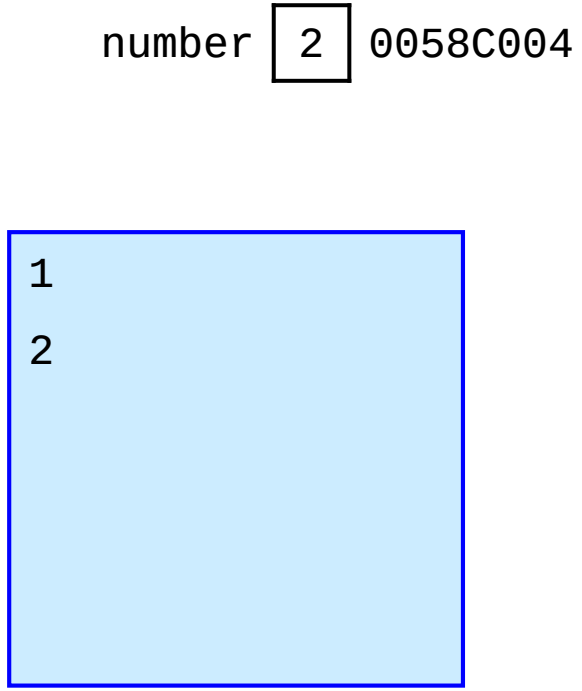
```
int main()
{
   useStatic();
   useStatic();
}

void useStatic()
{
   static int number = 1;
   cout << number << endl;
   number++;
   cout << number << endl;
}
```



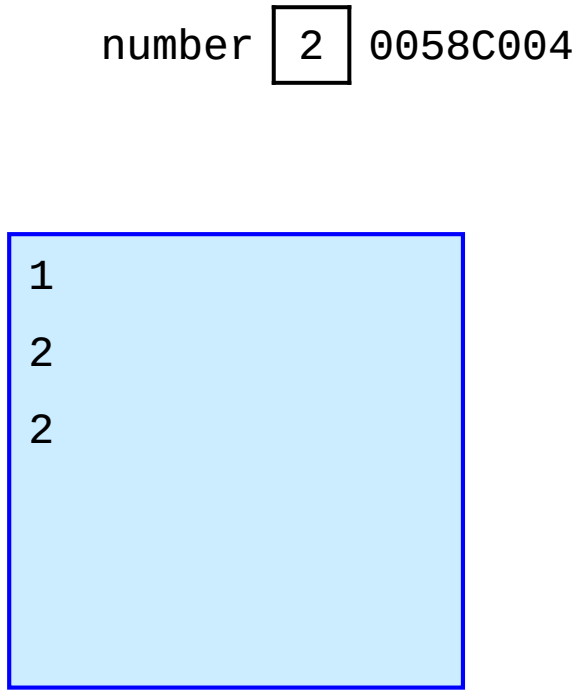
```
int main()
{
   useStatic();
   useStatic();
}

void useStatic()
{
   static int number = 1;
   cout << number << endl;
   number++;
   cout << number << endl;
}
```



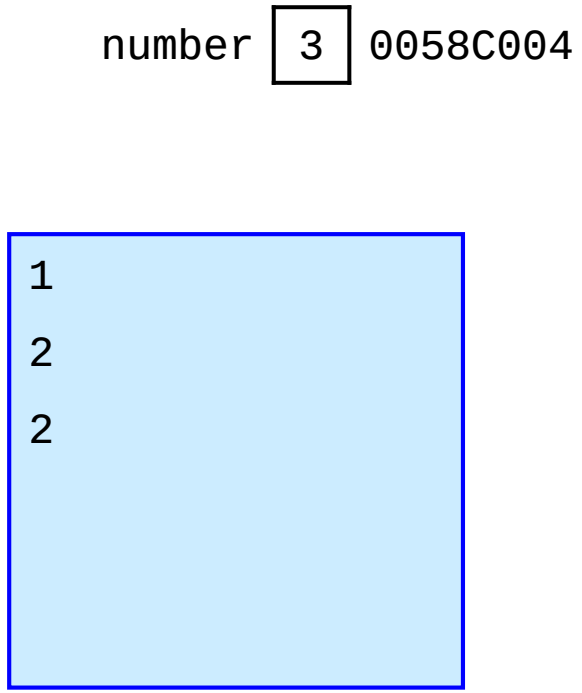
```
int main()
{
   useStatic();
   useStatic();
}

void useStatic()
{
   static int number = 1;
   cout << number << endl;
   number++;
   cout << number << endl;
}
```



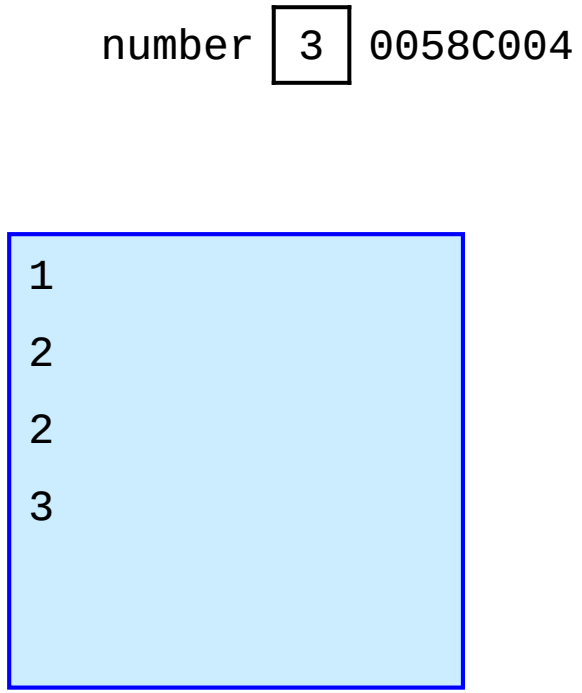
```
int main()
{
   useStatic();
   useStatic();
}

void useStatic()
{
   static int number = 1;
   cout << number << endl;
   number++;
   cout << number << endl;
}
```



```
int main()
{
   useStatic();
   useStatic();
}

void useStatic()
{
   static int number = 1;
   cout << number << endl;
   number++;
   cout << number << endl;
}
```



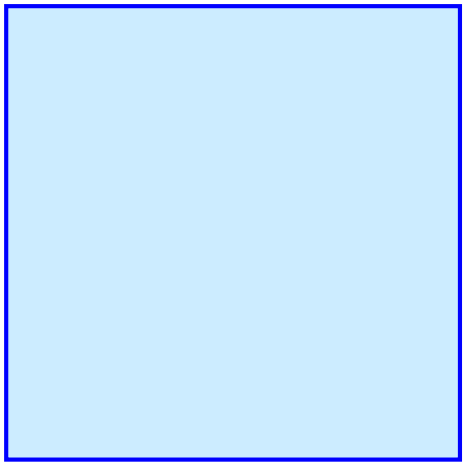
```
int main()
{
   useStatic();
   useStatic();
}

void useStatic()
{
   static int number = 1;
   cout << number << endl;
   number++;
   cout << number << endl;
}
```



```
int main()
{
   useStatic();
   useStatic();
}

void useStatic()
{
   static int number = 1;
   cout << number << endl;
   number++;
   cout << number << endl;
}
```

```
1
2
2
3
```

```
int main()
{
   useStatic();
   useStatic();
}

void useStatic()
{
   static int number;
   cout << number << endl;
   number++;
   cout << number << endl;
}
```



```
int main()
{
   useStatic();
   useStatic();
}

void useStatic()
{
   static int number;
   cout << number << endl;
   number++;
   cout << number << endl;
}
```

```
0
1
1
2
```

```
int main()
{
   useStaticArray();
   useStaticArray();
}

void useStaticArray()
{
   static int numbers[ 3 ] = { 1, 2, 3 };

   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "  ";
   cout << endl << endl;

   for( int i = 0; i < 3; i++ )
      numbers[ i ]++;

   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "  ";
   cout << endl << endl;
}
```

1 2 3

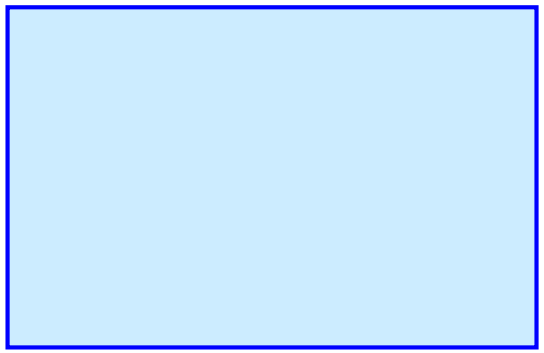
2 3 4

2 3 4

3 4 5

```
void useStaticArray();

int main()
{
   useStaticArray();
   useStaticArray();
}

void useStaticArray()
{
   static int numbers[ 3 ] = { 1, 2, 3 };
   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "  ";
   cout << endl << endl;

   for( int i = 0; i < 3; i++ )
      numbers[ i ]++;

   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "  ";
   cout << endl << endl;
}
```

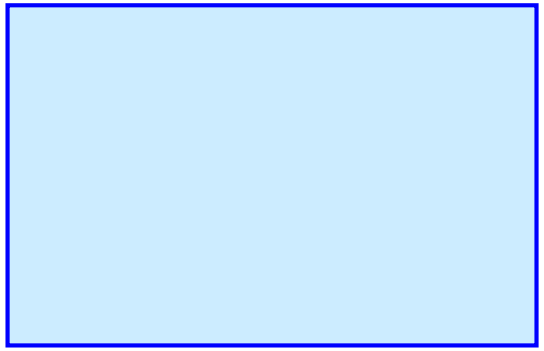
arraySize | 3 | 003E9B7C



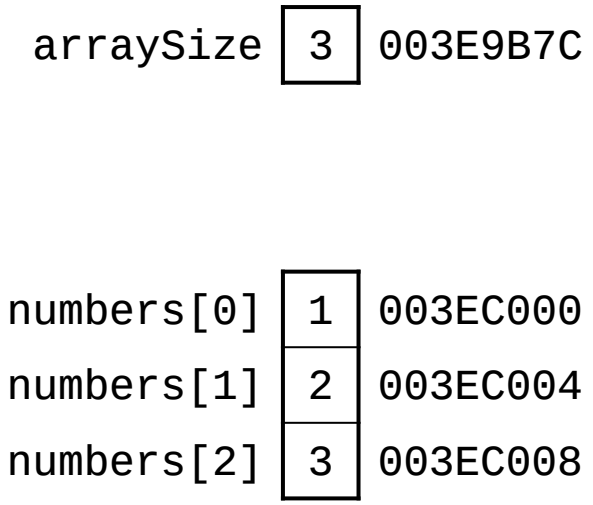
```
void useStaticArray();

int main()
{
   useStaticArray();
   useStaticArray();
}

void useStaticArray()
{
   static int numbers[ 3 ] = { 1, 2, 3 };
   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "  ";
   cout << endl << endl;

   for( int i = 0; i < 3; i++ )
      numbers[ i ]++;

   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "  ";
   cout << endl << endl;
}
```

arraySize	3	003E9B7C

numbers[0]	1	003EC000
numbers[1]	2	003EC004
numbers[2]	3	003EC008

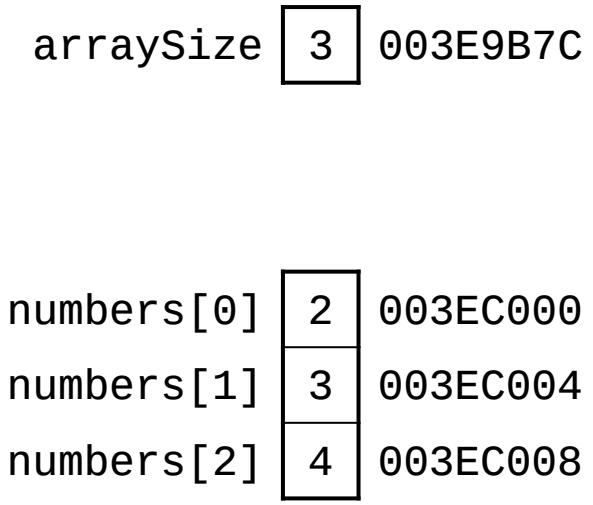


```
void useStaticArray();

int main()
{
   useStaticArray();
   useStaticArray();
}

void useStaticArray()
{
   static int numbers[ 3 ] = { 1, 2, 3 };
   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "  ";
   cout << endl << endl;

   for( int i = 0; i < 3; i++ )
      numbers[ i ]++;

   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "  ";
   cout << endl << endl;
}
```



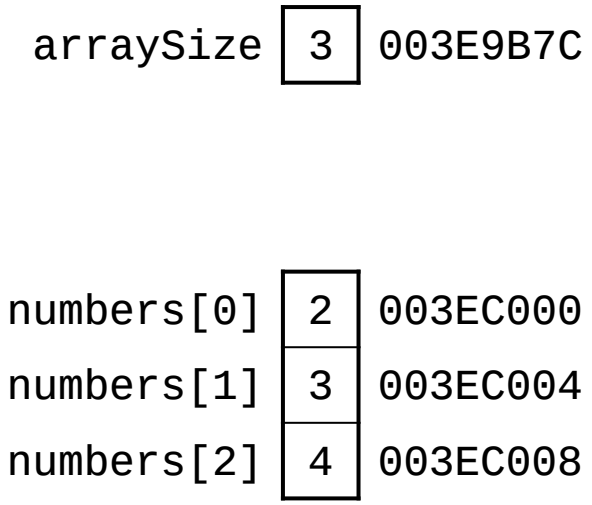

```
1  2  3
```

```
void useStaticArray();

int main()
{
   useStaticArray();
   useStaticArray();
}

void useStaticArray()
{
   static int numbers[ 3 ] = { 1, 2, 3 };
   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "  ";
   cout << endl << endl;

   for( int i = 0; i < 3; i++ )
      numbers[ i ]++;

   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "  ";
   cout << endl << endl;
}
```



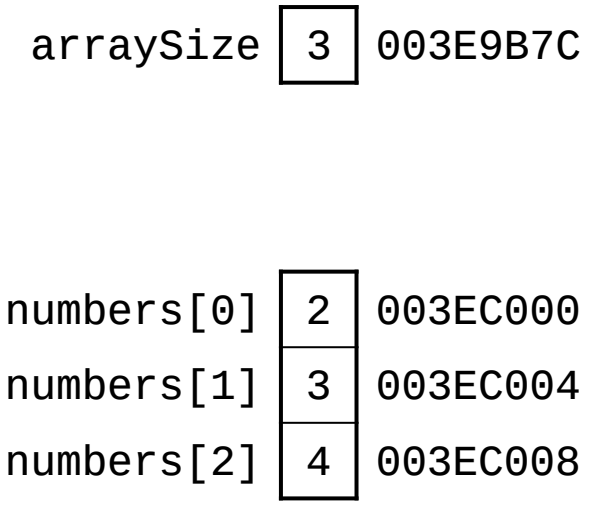

```
1  2  3
```

```
void useStaticArray();

int main()
{
   useStaticArray();
   useStaticArray();
}

void useStaticArray()
{
   static int numbers[ 3 ] = { 1, 2, 3 };
   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "  ";
   cout << endl << endl;

   for( int i = 0; i < 3; i++ )
      numbers[ i ]++;

   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "  ";
   cout << endl << endl;
}
```



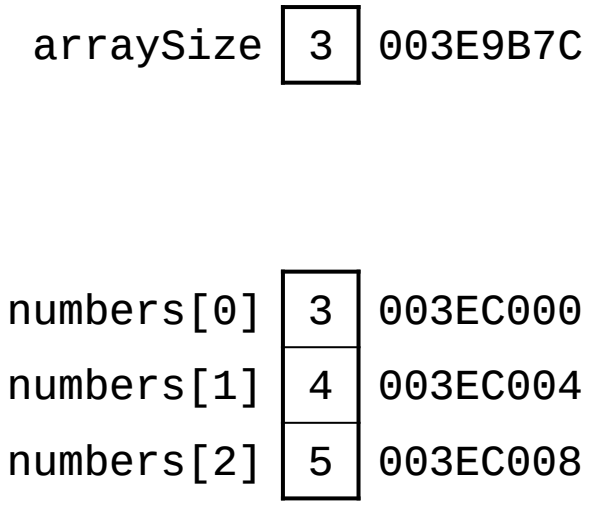

```
1  2  3
2  3  4
```

```
void useStaticArray();

int main()
{
   useStaticArray();
   useStaticArray();
}

void useStaticArray()
{
   static int numbers[ 3 ] = { 1, 2, 3 };
   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "  ";
   cout << endl << endl;

   for( int i = 0; i < 3; i++ )
      numbers[ i ]++;

   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "  ";
   cout << endl << endl;
}
```



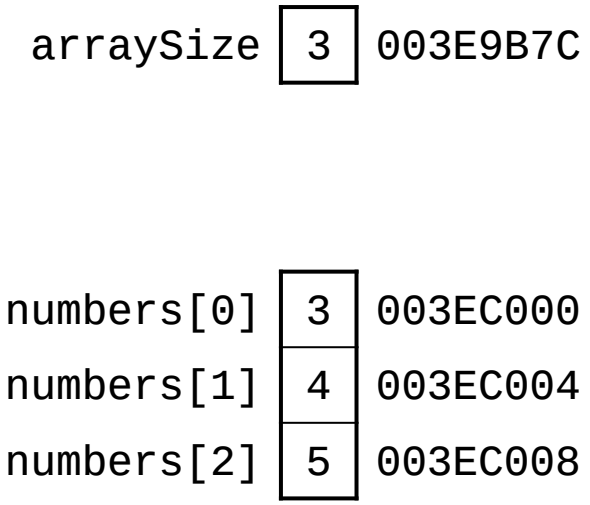

```
1  2  3
2  3  4
2  3  4
```

```
void useStaticArray();

int main()
{
   useStaticArray();
   useStaticArray();
}

void useStaticArray()
{
   static int numbers[ 3 ] = { 1, 2, 3 };
   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "  ";
   cout << endl << endl;

   for( int i = 0; i < 3; i++ )
      numbers[ i ]++;

   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "  ";
   cout << endl << endl;
}
```




```
1  2  3
2  3  4
2  3  4
```

```
void useStaticArray();

int main()
{
   useStaticArray();
   useStaticArray();
}

void useStaticArray()
{
   static int numbers[ 3 ] = { 1, 2, 3 };
   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "  ";
   cout << endl << endl;

   for( int i = 0; i < 3; i++ )
      numbers[ i ]++;

   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "  ";
   cout << endl << endl;
}
```

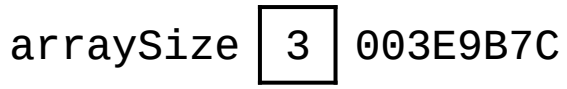


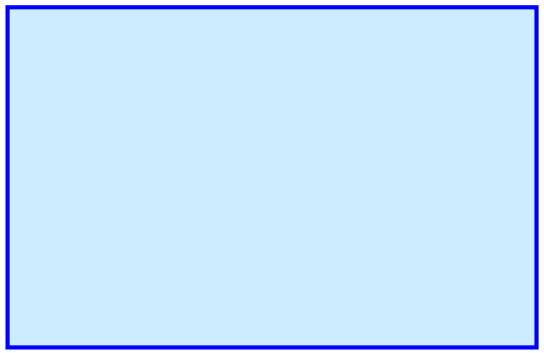

```
1  2  3
2  3  4
2  3  4
3  4  5
```

```
int main()
{
   useStaticArray();
   useStaticArray();
}

void useStaticArray()
{
   static int numbers[ 3 ];
   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "  ";
   cout << endl << endl;

   for( int i = 0; i < 3; i++ )
      numbers[ i ]++;

   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "  ";
   cout << endl << endl;
}
```

```
0  0  0

1  1  1

1  1  1

2  2  2
```

```
int main()
{
   useStaticArray();
   useStaticArray();
}

void useStaticArray()
{
   static int numbers[ 3 ];
   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "   ";
   cout << endl << endl;

   for( int i = 0; i < 3; i++ )
      numbers[ i ]++;

   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "   ";
   cout << endl << endl;
}
```

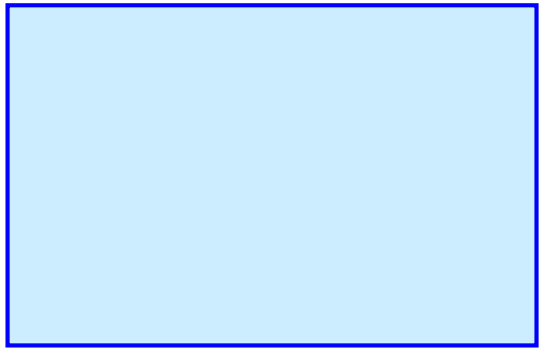




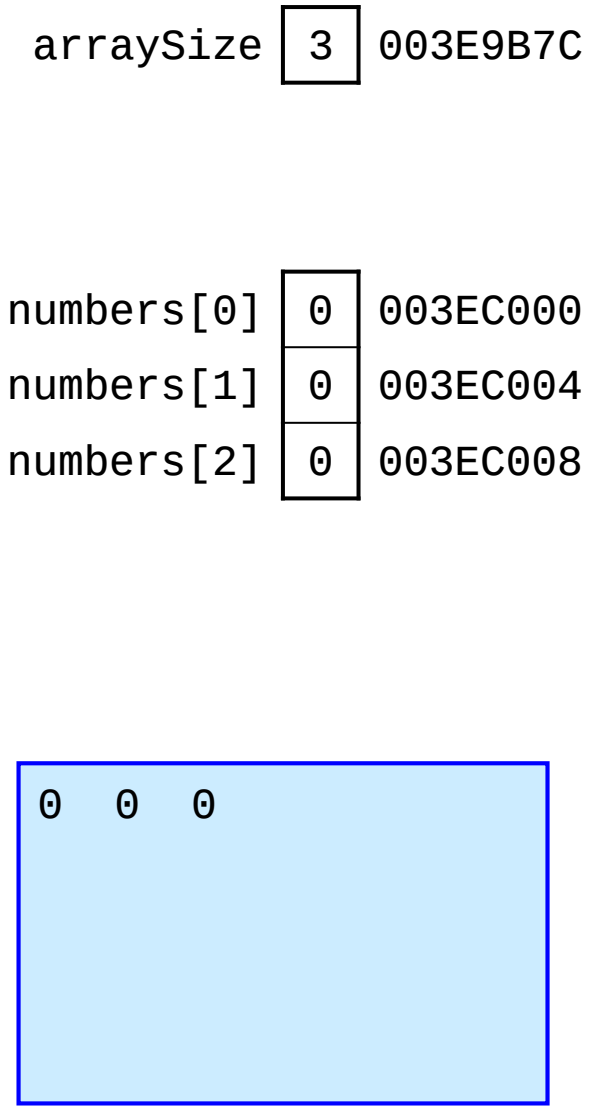
```
int main()
{
   useStaticArray();
   useStaticArray();
}

void useStaticArray()
{
   static int numbers[ 3 ];
   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "   ";
   cout << endl << endl;

   for( int i = 0; i < 3; i++ )
      numbers[ i ]++;

   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "   ";
   cout << endl << endl;
}
```

arraySize	3	003E9B7C

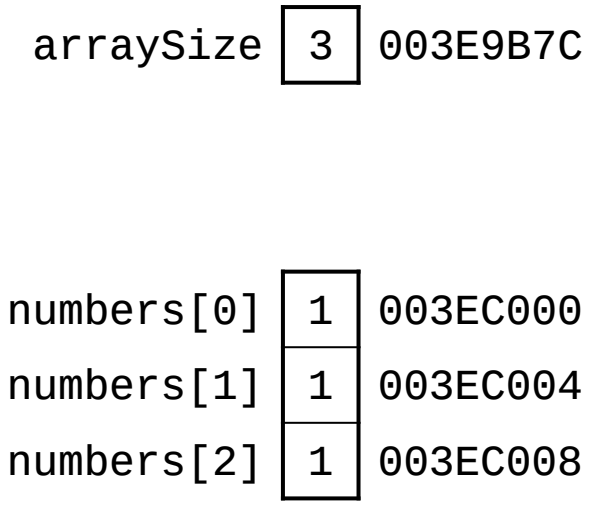
numbers[0]	0	003EC000
numbers[1]	0	003EC004
numbers[2]	0	003EC008

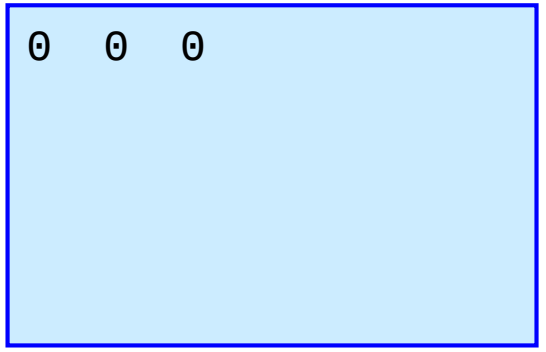


```
int main()
{
   useStaticArray();
   useStaticArray();
}

void useStaticArray()
{
   static int numbers[ 3 ];
   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "   ";
   cout << endl << endl;

   for( int i = 0; i < 3; i++ )
      numbers[ i ]++;

   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "   ";
   cout << endl << endl;
}
```

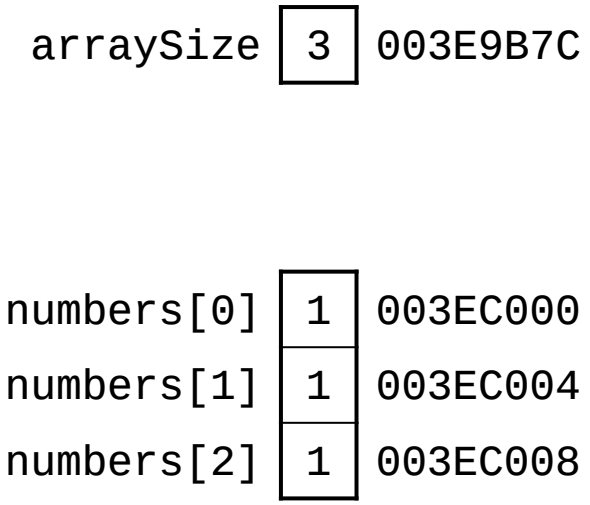


```
int main()
{
   useStaticArray();
   useStaticArray();
}

void useStaticArray()
{
   static int numbers[ 3 ];
   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "  ";
   cout << endl << endl;

   for( int i = 0; i < 3; i++ )
      numbers[ i ]++;

   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "  ";
   cout << endl << endl;
}
```



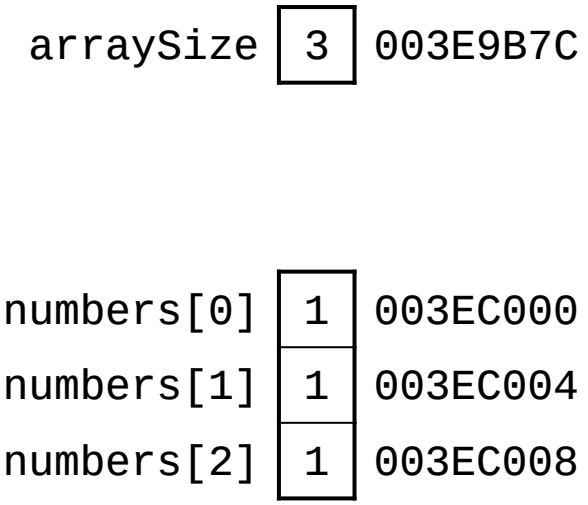


```
int main()
{
   useStaticArray();
   useStaticArray();
}

void useStaticArray()
{
   static int numbers[ 3 ];
   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "   ";
   cout << endl << endl;

   for( int i = 0; i < 3; i++ )
      numbers[ i ]++;

   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "   ";
   cout << endl << endl;
}
```



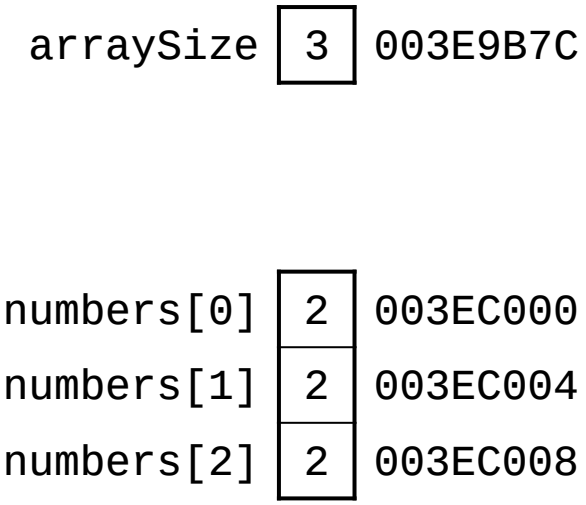

```
0   0   0
1   1   1
```

```
int main()
{
   useStaticArray();
   useStaticArray();
}

void useStaticArray()
{
   static int numbers[ 3 ];
   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "   ";
   cout << endl << endl;

   for( int i = 0; i < 3; i++ )
      numbers[ i ]++;

   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "   ";
   cout << endl << endl;
}
```



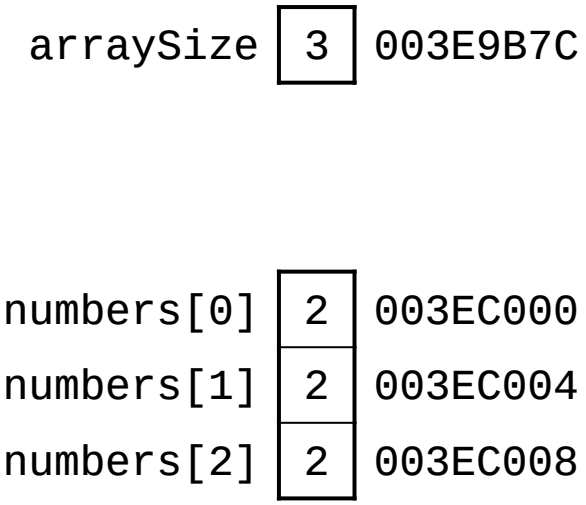

```
0   0   0
1   1   1
1   1   1
```

```
int main()
{
   useStaticArray();
   useStaticArray();
}

void useStaticArray()
{
   static int numbers[ 3 ];
   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "   ";
   cout << endl << endl;

   for( int i = 0; i < 3; i++ )
      numbers[ i ]++;

   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "   ";
   cout << endl << endl;
}
```




```
0   0   0
1   1   1
1   1   1
```

```
int main()
{
   useStaticArray();
   useStaticArray();
}

void useStaticArray()
{
   static int numbers[ 3 ];
   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "  ";
   cout << endl << endl;

   for( int i = 0; i < 3; i++ )
      numbers[ i ]++;

   for( int i = 0; i < 3; i++ )
      cout << numbers[ i ] << "  ";
   cout << endl << endl;
}
```




```
0  0  0
1  1  1
1  1  1
2  2  2
```

Two-dimensional Arrays

Matrix

a_{00}	a_{01}	a_{02}	a_{03}
a_{10}	a_{11}	a_{12}	a_{13}
a_{20}	a_{21}	a_{22}	a_{23}

- `int a[3][4];`
- `a` is an array of 3 one-dimensional arrays `a[0]`, `a[1]` and `a[2]`
- `a[0]` is an array of 4 elements `a[0][0]`, `a[0][1]`, `a[0][2]`, `a[0][3]`

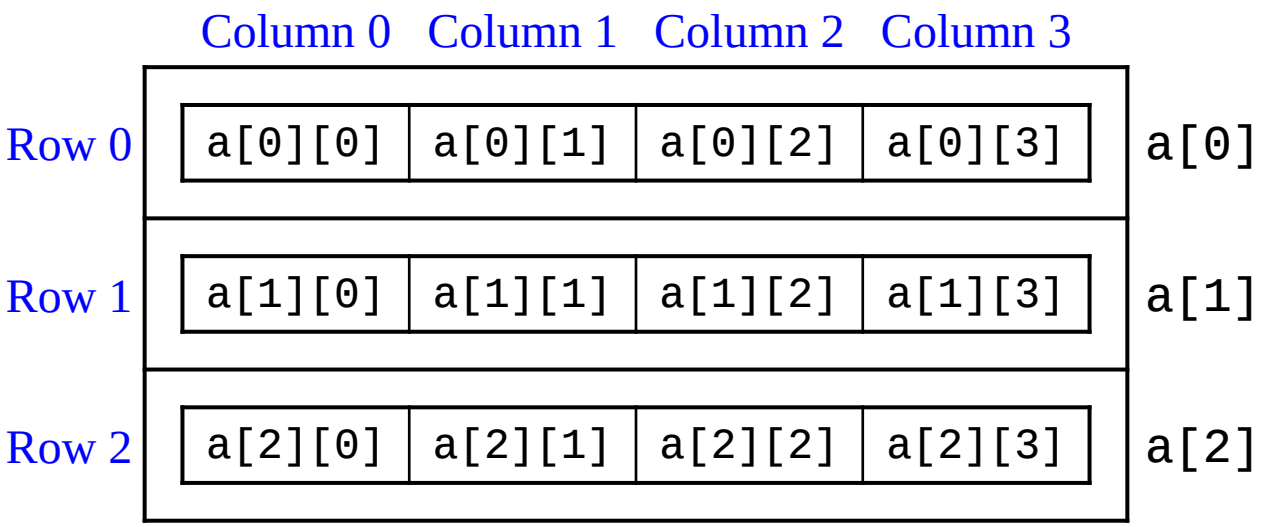
	Column 0	Column 1	Column 2	Column 3
Row 0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 2	a[2][0]	a[2][1]	a[2][2]	a[2][3]

Two-dimensional Arrays

Matrix

a_{00}	a_{01}	a_{02}	a_{03}
a_{10}	a_{11}	a_{12}	a_{13}
a_{20}	a_{21}	a_{22}	a_{23}

- `int a[3][4];`
- `a` is an array of 3 one-dimensional arrays `a[0]`, `a[1]` and `a[2]`
- `a[0]` is an array of 4 elements `a[0][0]`, `a[0][1]`, `a[0][2]`, `a[0][3]`



```
const int rows = 2;
const int columns = 3;
void printArray( const int data[][ columns ] );

int main()
{
   int data1[ rows ][ columns ] = { { 1, 2, 3 }, { 4, 5, 6 } };
   int data2[ rows ][ columns ] = { 1, 2, 3, 4, 5 };
   int data3[ rows ][ columns ] = { { 1, 2 }, { 4 } };

   printArray( data1 );
   printArray( data2 );
   printArray( data3 );
}

void printArray( const int data[][ columns ] )
{
   for ( int i = 0; i < rows; i++ )
   {
      for ( int j = 0; j < columns; j++ )
         cout << data[ i ][ j ] << ' ';
      cout << endl;
   }
   cout << endl;
}
```

data1

1	2	3
4	5	6

data2

1	2	3
4	5	0

data3

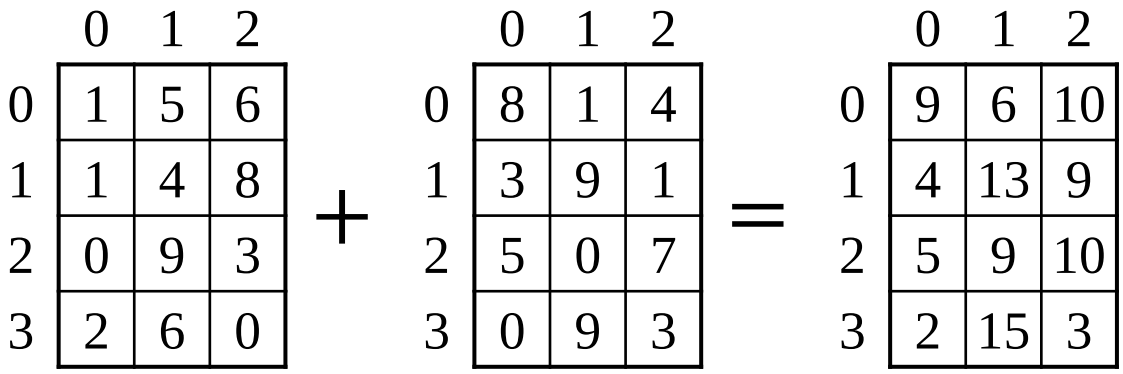
1	2	0
4	0	0

```
1 2 3
4 5 6

1 2 3
4 5 0

1 2 0
4 0 0
```

Matrix Addition

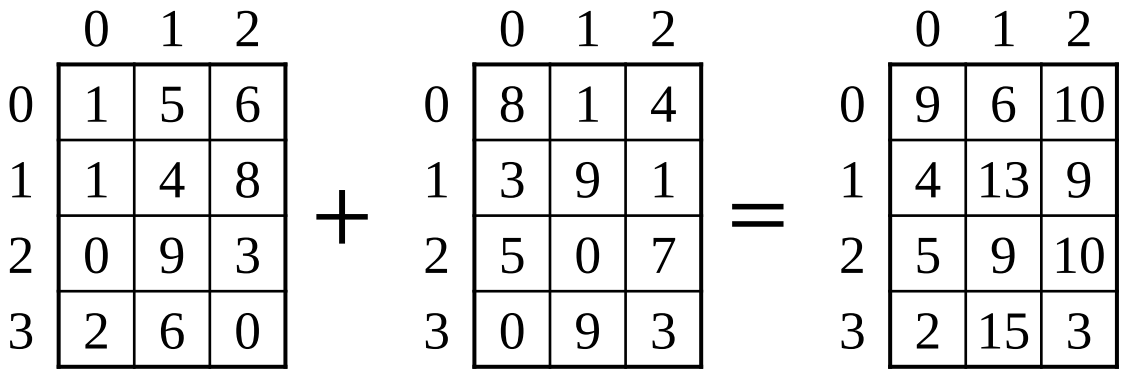
```
int main()
{
   int addend[ 10 ][ 10 ] = {};
   int adder[ 10 ][ 10 ] = {};
   int sum[ 10 ][ 10 ] = {};

   int numRows = 4;
   int numColumns = 3;

   genMatrix( addend, numRows, numColumns );
   genMatrix( adder, numRows, numColumns );

   addition( addend, adder, sum, numRows, numColumns );

   cout << "The sum of\n\n";
   display( addend, numRows, numColumns );
   cout << "and\n\n";
   display( adder, numRows, numColumns );
   cout << "is\n\n";
   display( sum, numRows, numColumns );
}
```

Matrix Addition



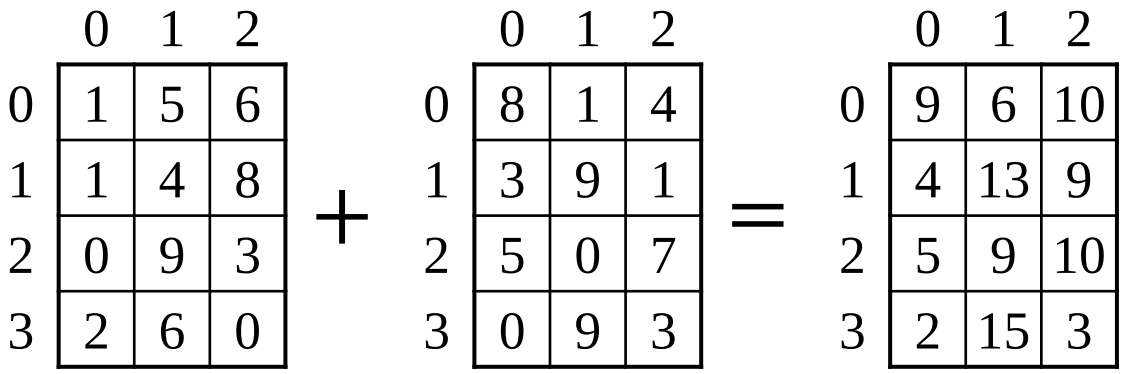

```
void addition( int addend[][ 10 ], int adder[][ 10 ], int sum[][ 10 ],
               int numRows, int numColumns )
{
   for(                                    )
      for(                                    )
                                                ;
}
```

Matrix Addition




```
void addition( int addend[][ 10 ], int adder[][ 10 ], int sum[][ 10 ],
               int numRows, int numColumns )
{
   for( int row = 0; row < numRows; row++ )
      for( int col = 0; col < numColumns; col++ )
                                                                    ;
}
```

Matrix Addition




```
void addition( int addend[][ 10 ], int adder[][ 10 ], int sum[][ 10 ],
               int numRows, int numColumns )
{
   for( int row = 0; row < numRows; row++ )
      for( int col = 0; col < numColumns; col++ )
         sum[ row ][ col ] = addend[ row ][ col ] + adder[ row ][ col ];
}
```

```
void genMatrix( int matrix[][ 10 ], int numRows, int numColumns )
{
   for( int row = 0; row < numRows; row++ )
      for( int col = 0; col < numColumns; col++ )
         matrix[ row ][ col ] = rand() % 10;
}

void addition( int addend[][ 10 ], int adder[][ 10 ], int sum[][ 10 ],
               int numRows, int numColumns )
{
   for( int row = 0; row < numRows; row++ )
      for( int col = 0; col < numColumns; col++ )
         sum[ row ][ col ] = addend[ row ][ col ] + adder[ row ][ col ];
}

void display( int matrix[][ 10 ], int numRows, int numColumns )
{
   for( int row = 0; row < numRows; row++ )
   {
      for( int col = 0; col < numColumns; col++ )
         cout << setw( 3 ) << matrix[ row ][ col ];
      cout << endl;
   }
   cout << endl;
}
```

Matrix Multiplication

```
int main()
{
   srand( static_cast< int >( time( 0 ) ) );
   int multiplicand[ 10 ][ 10 ];
   int multiplier[ 10 ][ 10 ];
   int product[ 10 ][ 10 ] = { 0 };
   int m = 4;
   int p = 3;
   int n = 5;

   genMatrix( multiplicand, m, p );
   genMatrix( multiplier, p, n );
   multiplication( multiplicand, multiplier, product, m, p, n );

   cout << "The product of\n\n";
   display( multiplicand, m, p );
   cout << "and\n\n";
   display( multiplier, p, n );
   cout << "is\n\n";
   display( product, m, n );
}
```

Matrix Multiplication

	0	1	2
0	1	1	0
1	2	0	1
2	3	1	1
3	0	2	1

*

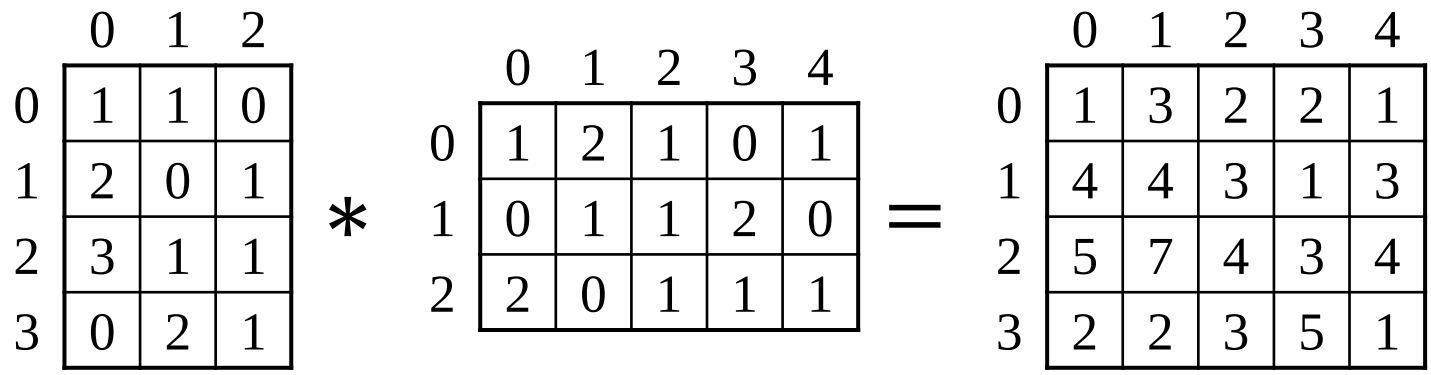
	0	1	2	3	4
0	1	2	1	0	1
1	0	1	1	2	0
2	2	0	1	1	1

=

	0	1	2	3	4
0	1	3	2	2	1
1	4	4	3	1	3
2	5	7	4	3	4
3	2	2	3	5	1

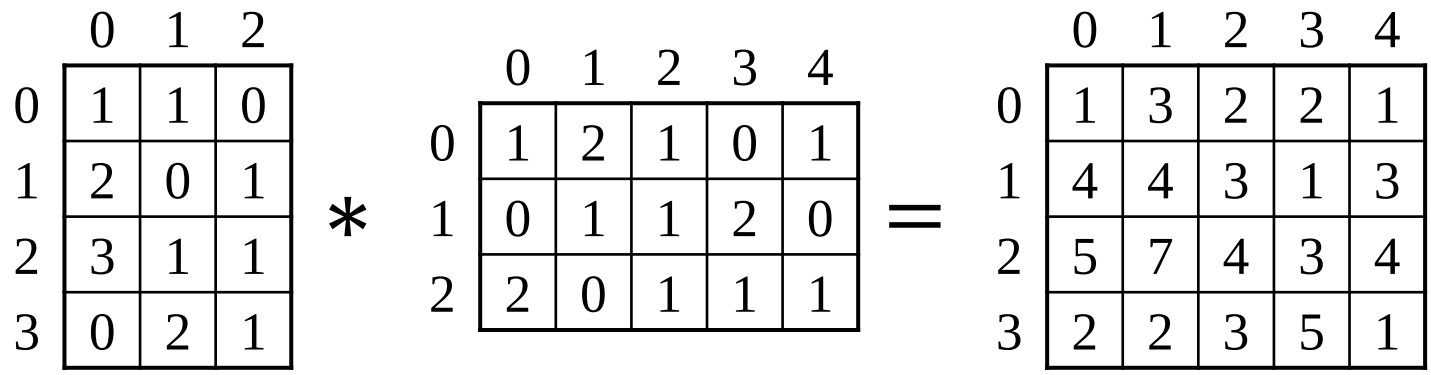
```
void multiplication( int multiplicand[][ 10 ], int multiplier[][ 10 ],
                     int product[][ 10 ], int m, int p, int n )
{
   for(                          )
      for(                          )
         for(                          )
                                                                    ;
}
```

Matrix Multiplication




```
void multiplication( int multiplicand[][ 10 ], int multiplier[][ 10 ],
                     int product[][ 10 ], int m, int p, int n )
{
   for( int i = 0; i < m; i++ )
      for( int j = 0; j < n; j++ )
         for( int k = 0; k < p; k++ )
                                                                    ;
}
```

Matrix Multiplication




```
void multiplication( int multiplicand[][ 10 ], int multiplier[][ 10 ],
                     int product[][ 10 ], int m, int p, int n )
{
   for( int i = 0; i < m; i++ )
      for( int j = 0; j < n; j++ )
         for( int k = 0; k < p; k++ )
            product[ i ][ j ] += multiplicand[ i ][ k ] * multiplier[ k ][ j ];
}
```

```
void genMatrix( int matrix[][ 10 ], int numRows, int numColumns )
{
   for( int row = 0; row < numRows; row++ )
      for( int col = 0; col < numColumns; col++ )
         matrix[ row ][ col ] = rand() % 5;
}

void multiplication( int multiplicand[][ 10 ], int multiplier[][ 10 ],
                     int product[][ 10 ], int m, int p, int n )
{
   for( int i = 0; i < m; i++ )
      for( int j = 0; j < n; j++ )
         for( int k = 0; k < p; k++ )
            product[ i ][ j ] += multiplicand[ i ][ k ] * multiplier[ k ][ j ];
}

void display( int matrix[][ 10 ], int numRows, int numColumns )
{
   for( int row = 0; row < numRows; row++ )
   {
      for( int col = 0; col < numColumns; col++ )
         cout << setw( 3 ) << matrix[ row ][ col ];
      cout << endl;
   }
   cout << endl;
}
```

Compute Sums of Rows

```
int main()
{
   srand( static_cast< int >( time( 0 ) ) );

   int matrix[ 10 ][ 10 ] = {};
   int numRows = 4;
   int numColumns = 3;

   genMatrix( matrix, numRows, numColumns );
   display( matrix, numRows, numColumns );

   cout << "Sums of rows:\n\n";
   for( int r = 0; r < numRows; r++ )
      cout << rowSum( matrix, r , numColumns ) << endl;
   cout << endl;
}

int rowSum( int matrix[][ 10 ], int r, int numColumns )
{
   int sum = 0;
   for( int c = 0; c < numColumns; c++ )
      sum += matrix[ r ][ c ];
   return sum;
}
```

0	12
1	13
2	12
3	8

	0	1	2
0	1	5	6
1	1	4	8
2	0	9	3
3	2	6	0

Compute Sums of Columns

```
int main()
{
   srand( static_cast< int >( time( 0 ) ) );

   int matrix[ 10 ][ 10 ] = {};
   int numRows = 4;
   int numColumns = 3;

   genMatrix( matrix, numRows, numColumns );
   display( matrix, numRows, numColumns );

   cout << "Sums of columns:\n\n";
   for( int c = 0; c < numColumns; c++ )
      cout << setw( 3 ) << columnSum( matrix, c, numRows );
   cout << endl << endl;
}

int columnSum( int matrix[][ 10 ], int c, int numRows )
{
   int sum = 0;
   for( int r = 0; r < numRows; r++ )
      sum += matrix[ r ][ c ];
   return sum;
}
```

	0	1	2
0	1	5	6
1	1	4	8
2	0	9	3
3	2	6	0

0	1	2
4	24	17

Compute Sums of Rows

```
int main()
{
   srand( static_cast< int >( time( 0 ) ) );

   int matrix[ 10 ][ 10 ] = {};
   int numRows = 4;
   int numColumns = 3;

   genMatrix( matrix, numRows, numColumns );
   display( matrix, numRows, numColumns );

   cout << "Sums of rows:\n\n";
   for( int r = 0; r < numRows; r++ )
      cout << rowSum( matrix[ r ], numColumns ) << endl;
   cout << endl;
}

int rowSum( int       , int numColumns )
{
   int sum = 0;
   for( int c = 0; c < numColumns; c++ )
      sum +=          ;
   return sum;
}
```

0	12
1	13
2	12
3	8

	0	1	2
0	1	5	6
1	1	4	8
2	0	9	3
3	2	6	0

```
int main()
{
   srand( static_cast< int >( time( 0 ) ) );

   int matrix[ 10 ][ 10 ] = {};
   int numRows = 4;
   int numColumns = 3;

   genMatrix( matrix, numRows, numColumns );
   display( matrix, numRows, numColumns );

   cout << "Sums of rows:\n\n";
   for( int r = 0; r < numRows; r++ )
      cout << rowSum( matrix[ r ], numColumns ) << endl;
   cout << endl;
}

int rowSum( int row[], int numColumns )
{
   int sum = 0;
   for( int c = 0; c < numColumns; c++ )
      sum += row[ c ];
   return sum;
}
```

0	12
1	13
2	12
3	8

	0	1	2
0	1	5	6
1	1	4	8
2	0	9	3
3	2	6	0

Introduction to C++ Standard Library Class Template `vector`

```
int main()
{
   vector< int > v1( 3 );
   vector< int > v2( 6 );

   cout << "Size of v1 is " << v1.size() << "\nv1:";
   output( v1 );

   cout << "Size of v2 is " << v2.size() << "\nv2:";
   output( v2 );

   for( size_t i = 0; i < 3; i++ )
      v1[ i ] = i + 1;

   for( size_t i = 0; i < 6; i++ )
      v2[ i ] = i + 4;

   cout << "v1:";
   output( v1 );
   cout << "v2:";
   output( v2 );

   vector< int > v( v1 );

   cout << "Size of v is " << v.size() << "\nv:";
   output( v );
```

```
    if( v1 != v2 )
       cout << "v1 != v2\n\n";

    cout << "v1 = v2:\n\n";
    v1 = v2;

    cout << "v1:";
    output( v1 );
    cout << "v2:";
    output( v2 );

    if( v1 == v2 )
       cout << "v1 == v2\n\n";

    cout << "v1[ 3 ] is " << v1[ 3 ] << endl << endl;

    v1[ 3 ] = 100;

    cout << "v1[ 3 ] is " << v1[ 3 ] << endl << endl;

    cout << "v1:";
    output( v1 );
}
```

```
Size of v1 is 3
v1:   0   0   0

Size of v2 is 6
v2:   0   0   0   0   0   0

v1:   1   2   3

v2:   4   5   6   7   8   9

Size of v is 3
v:   1   2   3

v1 != v2

v1 = v2:

v1:   4   5   6   7   8   9

v2:   4   5   6   7   8   9

v1 == v2
```

```
v1[ 3 ] is 7

v1[ 3 ] is 100

v1:   4   5   6 100   8   9
```

Introduction to C++ Standard Library Class string

```
int main()
{
   string s1( "happy" );
   string s2( " birthday" );
   string s3;
   string s4;

   cout << "Enter a string: ";
   cin >> s4;

   cout << "s1 is \"" << s1 << "\"; s2 is \"" << s2
      << "\"; s3 is \"" << s3 << "\"; s4 is \"" << s4 << "\"\n\n";

   cout << "s1 length " << s1.size() << "; s2 length " << s2.size()
      << "; s3 length " << s3.size() << "; s4 length " << s4.size();

   cout << "\n\nThe results of comparing s2 and s1:"
      << "\ns2 == s1 yields " << ( s2 == s1 ? "true" : "false" )
      << "\ns2 != s1 yields " << ( s2 != s1 ? "true" : "false" )
      << "\ns2 >  s1 yields " << ( s2 > s1 ? "true" : "false" )
      << "\ns2 <  s1 yields " << ( s2 < s1 ? "true" : "false" )
      << "\ns2 >= s1 yields " << ( s2 >= s1 ? "true" : "false" )
      << "\ns2 <= s1 yields " << ( s2 <= s1 ? "true" : "false" );
```

```
cout << "\n\nTesting s3.empty():" << endl;

if ( s3.empty() )
{
   cout << "s3 is empty; assigning s1 to s3;" << endl;
   s3 = s1;
   cout << "s3 is \"" << s3 << "\"";
}

cout << "\n\nAfter s1 += s2, s1 is ";
s1 += s2;
cout << s1;

cout << "\n\ns1 += \" to you\" yields" << endl;
s1 += " to you";
cout << "s1 is " << s1 << "\n\n";

cout << "The substring of s1 starting at location 0 for\n"
   << "14 characters, s1.substr(0, 14), is:\n"
   << s1.substr( 0, 14 ) << "\n\n";

cout << "The substring of s1 starting at\n"
   << "location 15, s1.substr(15), is:\n"
   << s1.substr( 15 ) << endl;
```

```
   string s5( s1 );
   cout << "\ns5 is " << s5;

   s1[ 0 ] = 'H';
   s1[ 6 ] = 'B';
   cout << "\n\ns1 after s1[0] = 'H' and s1[6] = 'B' is: " << s1;

   cout << "\n\ns1[0] is " << s1[ 0 ] << "; s1[2] is " << s1 [ 2 ]
      << "; s1[s1.size()-1] is " << s1[ s1.size() - 1 ];

   cout << "\n\nAttempt to assign 'd' to s1.at( 30 ) yields:" << endl;
   s1.at( 30 ) = 'd';
}
```

```
Enter a string: Using class string
s1 is "happy"; s2 is " birthday"; s3 is ""; s4 is "Using"

s1 length 5; s2 length 9; s3 length 0; s4 length 5

The results of comparing s2 and s1:
s2 == s1 yields false
s2 != s1 yields true
s2 >  s1 yields false
s2 <  s1 yields true
s2 >= s1 yields false
s2 <= s1 yields true

Testing s3.empty():
s3 is empty; assigning s1 to s3;
s3 is "happy"

After s1 += s2, s1 is happy birthday

s1 += " to you" yields
s1 is happy birthday to you
```

```
The substring of s1 starting at location 0 for
14 characters, s1.substr(0, 14), is:
happy birthday

The substring of s1 starting at
location 15, s1.substr(15), is:
to you

s5 is happy birthday to you

s1 after s1[0] = 'H' and s1[6] = 'B' is: Happy Birthday to you

s1[0] is H; s1[2] is p; s1[s1.length()-1] is u

Attempt to assign 'd' to s1.at( 30 ) yields:
```

```
int main()
{
   string s1( "happy" );
   string s2( " birthday" );
   string s3;
   string s4;
   cout << "Enter a string: ";
   cin >> s4;
   cout << "s1 is \"" << s1 << "\"; s2 is \"" << s2
      << "\"; s3 is \"" << s3 << "\"; s4 is \"" << s4 << "\"\n\n";
```

```
int main()
{
   char s1[ 80 ] = "happy";
   char *s2 = " birthday";
   char s3[ 80 ] = "";
   char s4[ 80 ];
   cout << "Enter a string: ";
   cin >> s4;
   cout << "s1 is \"" << s1 << "\"; s2 is \"" << s2
      << "\"; s3 is \"" << s3 << "\"; s4 is \"" << s4 << "\"\n\n";
```

```
   cout << "s1 length " << s1.length() << "; s2 length " << s2.length()
      << "; s3 length " << s3.length() << "; s4 length " << s4.length();

   cout << "\n\nThe results of comparing s2 and s1:"
      << "\ns2 == s1 yields " << ( s2 == s1 ? "true" : "false" )
      << "\ns2 != s1 yields " << ( s2 != s1 ? "true" : "false" )
      << "\ns2 >  s1 yields " << ( s2 > s1 ? "true" : "false" )
      << "\ns2 <  s1 yields " << ( s2 < s1 ? "true" : "false" )
      << "\ns2 >= s1 yields " << ( s2 >= s1 ? "true" : "false" )
      << "\ns2 <= s1 yields " << ( s2 <= s1 ? "true" : "false" );
```

```
   cout << "s1 length " << strlen( s1 ) << "; s2 length " << strlen( s2 )
      << "; s3 length " << strlen( s3 ) << "; s4 length " << strlen( s4 );

   cout << "\n\nThe results of comparing s2 and s1:"
      << "\ns2 == s1 yields " << ( strcmp( s2, s1 ) == 0 ? "true" : "false" )
      << "\ns2 != s1 yields " << ( strcmp( s2, s1 ) != 0 ? "true" : "false" )
      << "\ns2 >  s1 yields " << ( strcmp( s2, s1 ) == 1 ? "true" : "false" )
      << "\ns2 <  s1 yields " << ( strcmp( s2, s1 ) == -1 ? "true" : "false" )
      << "\ns2 >= s1 yields " << ( strcmp( s2, s1 ) >= 0 ? "true" : "false" )
      << "\ns2 <= s1 yields " << ( strcmp( s2, s1 ) <= 0 ? "true" : "false" );
```

```
   cout << "\n\nTesting s3.empty():" << endl;
   if ( s3.empty() )
   {
      cout << "s3 is empty; assigning s1 to s3;" << endl;
      s3 = s1;
      cout << "s3 is \"" << s3 << "\"";
   }
   cout << "\n\nAfter s1 += s2, s1 is ";
   s1 += s2;
   cout << s1;
```

```
   cout << "\n\nTesting s3.empty():" << endl;
   if ( strcmp( s3, "" ) == 0 )
   {
      cout << "s3 is empty; assigning s1 to s3;" << endl;
      strcpy( s3, s1 );
      cout << "s3 is \"" << s3 << "\"";
   }
   cout << "\n\nAfter s1 += s2, s1 is ";
   strcat( s1, s2 );
   cout << s1;
```

```
   cout << "\n\ns1 += \" to you\" yields" << endl;
   s1 += " to you";
   cout << "s1 is " << s1 << "\n\n";

   cout << "The substring of s1 starting at location 0 for\n"
      << "14 characters, s1.substr(0, 14), is:\n"
      << s1.substr( 0, 14 ) << "\n\n";
```

```
   cout << "\n\ns1 += \" to you\" yields" << endl;
   strcat( s1, " to you" );
   cout << "s1 is " << s1 << "\n\n";

   char s6[ 16 ];
   strncpy( s6, s1, 15 );
   s6[ 15 ] = '\0';
   cout << "The substring of s1 starting at location 0 for\n"
      << "14 characters, s1.substr(0, 14), is:\n"
      << s6 << "\n\n";
```

```
   cout << "The substring of s1 starting at\n"
      << "location 15, s1.substr(15), is:\n"
      << s1.substr( 15 ) << endl;
```

```
   char s7[ 10 ];
   memcpy( s7, s1 + 15, strlen( s1 ) - 15 );
   s7[ strlen( s1 ) - 15 ] = '\0';
   cout << "The substring of s1 starting at\n"
      << "location 15, s1.substr(15), is:\n"
      << s7 << endl;
```

```
   string s5( s1 );
   cout << "\ns5 is " << s5;
   s1[ 0 ] = 'H';
   s1[ 6 ] = 'B';
   cout << "\n\ns1 after s1[0] = 'H' and s1[6] = 'B' is: " << s1;
   cout << "\n\ns1[0] is " << s1[ 0 ] << "; s1[2] is " << s1 [ 2 ]
      << "; s1[s1.length()-1] is " << s1[ s1.length() - 1 ];
   cout << "\n\nAttempt to assign 'd' to s1.at( 30 ) yields:\n";
}
```

```
   char s5[ 30 ];
   strcpy( s5, s1 );
   cout << "\ns5 is " << s5;
   s1[ 0 ] = 'H';
   s1[ 6 ] = 'B';
   cout << "\n\ns1 after s1[0] = 'H' and s1[6] = 'B' is: " << s1;
   cout << "\n\ns1[0] is " << s1[ 0 ] << "; s1[2] is " << s1 [ 2 ]
      << "; s1[s1.length()-1] is " << s1[ strlen( s1 ) - 1 ];
   cout << "\n\nAttempt to assign 'd' to s1.at( 30 ) yields:\n";
}
```